APPENDIX D

Input: A continuous variable x of dimension m x *1*
Output: 1. A flag indicating whether the input vector is exponentially distributed
H = 1 : yes; H = 0 : no
2. The mean value *meanv* and minimum value *minv* of the sample
Process:

```
n = 51                      // sample size
x1 = [0:1/(n-1):1]          // x1 is a vector of length n, from 0 to 1 with step 1/(n-1)
x2 = zeros(1, n)            // initialize a vector of zeros with the same length of x1
B = sorted(x)               // in ascending order
idx = m * x1
idx = round(idx)            // index of samples
i = 1
While (idx(i) == 0)
        idx(i) = 1
        i++
End While                   // make sure indexes are not out of bound
idx(n) = m                  // last sample is the maximum value
For i = 1:n
        x2(i) = B(idx(i))
End For                     //x2 is the vector of samples
minv = x2(1);               //first element is the minimum value
meanv = mean(x2);           //mean value of samples
//log-scale x2
For i = 1:n
```

Compute $x2(i) = 1 - e^{\frac{x2(i)-\text{minv}}{\text{minv}-meanv}}$

```
End For                     // if x2 now is uniform distributed, x is exponential distributed
//later is the KS test, test whether x1 and x2 have the "same" distribution
max_d = 0
For i = 1:n-1
    If (abs(x2(i) – x1(i)) > max_d)
            max_d = abs(x2(i) – x1(i))
    End If
    If (abs(x2(i) – x1(i+1)) > max_d)
            max_d = abs(x2(i) – x1(i+1))
    End If
End For
If (abs(x2(n) – x1(n)) > max_d)
    max_d = abs(x2(n) – x1(n))
End If
en = sqrt(n)
prob = probks((en + 0.12 + 0.11/en)*max_d)
```

```
If (prob > 0.3)
    H = 1
Else
    H = 0
End If
Return H, minv, meanv;
```

Sorting is done in ascending order.